Proc. VIIth Digital Image Computing: Techniques and Applications, Sun C., Talbot H., Ourselin S. and Adriaansen T. (Eds.), 10-12 Dec. 2003, Sydney

Compressing Sets of Similar Medical Images Using Multilevel Centroid Technique

Yasser El-Sonbaty[1], Meer Hamza[2], George Basily[3]

Dept. of Computer Eng., Arab Academy for Science & Technology, Alexandria 1029, EGYPT
[1] yasser@aast.edu, [2] mhamza@aast.edu, [3] mbawad@aast.edu

Abstract. Application areas such as medical imaging or satellite imaging often store large collections of similar images. Lossless compression techniques are usually needed in such critical applications. Previous researches have introduced the centroid method, which gets benefit from the inter-image redundancy ì *the set redundancy*î. In this paper a new algorithm is proposed as an extension of the centroid method. Experimental results with two sets of CT and MRI brain images demonstrate the efficiency and superiority of the proposed algorithm in respect to compression ratio.

1. Introduction

The field of digital image processing is continually evolving in two principle application areas: improvement of pictorial information for human interpretation such as in medical imaging, archeology and astronomy, and processing of scene data for autonomous machine perception such as in processing of fingerprints, screening of x-rays and blood samples.

Increasingly, medical images are acquired or stored digitally. This is especially true of grayscale images that are used in radiology applications. These images may be very large in size and number, hence compression offers a perfect mean to reduce the cost of storage and increase the speed of transmission [1]. In addition, medical images must be stored without any lose of information since the fidelity of the image is quite critical in diagnosis. This requires lossless compression techniques. Recent techniques for compressing images concentrate on how to reduce the redundancy presented in individual images. Whereas few research has focused on how to get benefit from redundancy between set of similar images which appears in many application areas such as in medical images inside a hospital database or in geographical information systems.

In this paper a new algorithm is proposed as an extension to the existing centroid algorithm [2] for compressing a set of similar medical images. The rest of this paper is organized as follows. Section 2 contains a review on related work. In section 3, the proposed algorithm is introduced. The experimental results on CT (Computed

Tomography) and MRI (Magnetic Resonance Imaging) data sets are presented in section 4. Finally, conclusions are discussed in section 5.

2. Related work

Approaches for compressing similar images can be classified into two main categories. The set redundancy category [2,8,9], which is based on predictive coding to get benefit from the inter-image redundancy and statistical correlation between images. While the second category defines different methods based on quadtrees for the representation and manipulation of clusters of images [10,11,13,14,15,16].

2.1 Set Redundancy Based Approaches

The term set redundancy were introduced in [17] which is defined as: the inter-image redundancy that exists in a set of similar images, and refers to the common information found in more than one image in the set. This can be similar pixel intensity in the same areas, comparable histograms, similar edge distributions, or analogous distributions of features. In order to incorporate the concept of set redundancy, a two- step procedure can be used. In the first step, the images are decorrelated from the set by extracting the set redundancy; in the second step, the images are compressed by using any compression method. The next sections introduce different compression methods based on set redundancy.

2.1.1 The Min-Max Differential (MMD) method

The MMD method [8] creates a ìminî and ìmaxî image from a set of similar images. To create the ìminî image, MMD chooses the smallest value across all images for every pixel position. Similarly, choosing the largest pixel value for every pixel position creates the ìmaxî image. Then, the MMD processes every image in the set by replacing the original pixel values with the differences from either the ìminî or the ìmaxî image (whichever is smaller). This operation reduces the dynamic range of pixel values, so that applying any standard entropy coder on the MMD-processed images can improve the results.

2.1.2 The Min-Max Predictive (MMP) method

The MMP method [9] also uses ìminî and ìmaxî images such as in the MMD method. The possible value of pixel P_i is assumed to lie between min_i and max_i. Neighboring pixels tend to fall in approximately the same area between the minimum and maximum values. The position of every pixel between its corresponding minimum and maximum values can be represented by ìlevelî L_i

$$L_i = N\,((P_i - min_i) / (max_i - min_i)) \quad (1)$$

Where N is the number of levels between the maximum and minimum values. Neighboring pixels usually have approximately the same ìlevelî, so the ìlevelî values have smaller variation than the pixel values. Therefore the ìlevelsî are better predictors for the next pixel values than the pixel values themselves. The MMP method predicts levels using the formula:

$$L_i = (L_{upper} + L_{left}) / 2 \quad (2)$$

Where L_{upper} is the level of the upper neighbor pixel and L_{left} is the level of the left neighbor. L_i, min_i, and max_i are used to predict a value for pixel P_i. The difference between the predicted value and the original value is stored and compressed using standard entropy coders.

2.1.3 Centroid method

The centroid method [2] depends on predictive decorrelation where an estimate of the image is obtained and then subtracted from the original image. If the prediction is efficient enough, the difference image will contain small values and has a laplacian distribution with most of the values very close to zero.

For a set of k images with N pixels per image, the formula for predicting the value of a pixel *i* in image *j* can be expressed as follows:

$$C_{i,j} = m_i \quad (3)$$

Where $C_{i,j}$ is the predicted value and m_i is the average value of position *i* across all images. This model is simple but not very efficient. A more advanced model is also proposed as follows:

$$C_{i+1,j} = m_{i+1} + x_{i,j} - m_i \quad (4)$$

$$D_{i+1,j} = x_{i+1,j} \text{ ñ } C_{i+1,j} \quad (5)$$

Where C_{i+1} is an estimate at position *i+1* in image *j*, $x_{i,j}$ is the pixel at position *i* in image *j*, m_i is the average value of position *i* across all images, and $D_{i+1,j}$ is the difference value of position *i+1* in image *j* between the original value and the predicted one. The detailed derivation of equations 4 and 5 is shown in [2]. Eq. (4) is so called the centroid method.

The problem with the minimum, maximum and average images is their sensitivity to outliers [17]. The median image can be used instead of the average image in the centroid method to reduce the influence of outliers. On the other hand, the set redundancy methods are fast, lossless, easy to implement, and can compress and decompress individual images from the set without requiring global calculations on the whole set.

2.2 Quadtree Based Approaches

The quadtree [4,5] is one of the widely used structures for image representation. This structure is efficient to store 2D images and has been frequently used in the field of computer graphics [6] and content-based image retrieval [7]. A quadtree is built by recursive division of the space in four quadrants or squares of the same size so that a node of the quadtree represents a quadrant. The root node represents the initial quadrant containing the whole image. The most widely known quadtree allows cutting an image in regions or quadrants according to a given criterion. If an image is not homogeneous (according to a particular criterion), the quadtree root has four descendant nodes representing the four first level image quadrants. A node is a leaf when its corresponding image quadrant is homogeneous; otherwise the node is internal [3]. In general, the quadtree is unbalanced.

2.2.1 Overlapping of hierarchical quadtrees

The overlapping mechanism is used to store images in sequence: when a new image *i* is inserted, its quadtree overlapps the last quadtree of the sequence, i.e. the quadtree of image *i-1,* if parts of the two image quadtrees have the same value. The quadtree *i* reference common nodes of quadtree *i-1.* Other different leaf nodes appear with their path from the root in quadtree *i*. This approach [14] uses pointers to reference nodes and can be extended to linear quadtrees by storing node identifiers in B^+ trees and apply overlapping mechanism to the B^+ trees.

In this approach, reading an image is as much time consuming as using independent quadtrees, i.e. one quadtree for each image. Any image insertion must be at the end of the sequence and any modification in any image quadtree leads to a new quadtree that must be stored at the end of the sequence. No quadtree deletion is permitted if its quadrants are referenced by other quadtrees. Using pointers to reference nodes is costly in managing memory space. Main areas of application are compaction and delivery of video.

2.2.2 Inverted Quadtrees

Two inverted quadtrees are proposed in [10,13], called Fully (*FI-Quadtree*) and Dynamic (*DI-Quadtree*). In those structures, a set of binary images is encoded in a single quadtree. The FI-Quadtree [10] consists of a full quadtree where each node has four children except the leaf nodes. Each node holds a bit string of maximum length *n* for representing *n* images. Each bit is associated with a separate image. A black node in the quadtree of any image is identified by a 1 in the bit corresponding to the image in the corresponding node of the full quadtree. On the other hand, in the DI-quadtree [13], each node of the full quadtree points to a list containing only the identifiers of the images that have the corresponding black node in their quadtree. In comparison, the DI-quadtree is dynamic because any number of images can be represented while in FI-quadtree, number of images is limited to the length of the node bit string.

2.2.3 Generic Quadtrees

The generic quadtrees [11] are based on two principles of sharing of quadrant values between images: *explicit* and *implicit.* Sharing is explicit if a quadrant value is stored and associated with all image identifiers that share the same value. Implicit sharing is built by constructing a tree structure called *Image Tree* where images are arranged in the tree according to their similarity. Image *i* implicitly shares a quadrant value with its parent image if it is not explicitly associated with another value. The generic quadtree is a single quadtree whose generic nodes represent the quadtree nodes of a cluster of images. Each generic node *n* contains the information needed to rebuild the value of the node *n* in each image quadtree. Each generic node can be seen as a table of two columns and one or more lines. Each line *l* of a generic node *n* contains a list of image identifiers and a value *v* of quadtree node. *v* is the value of node *n* in each image quadtree whose identifier *i* appears in line *l*. a generic node can take the following values, ì*I*î meaning that the node is internal ìit has four descendantsî; black if it is black leaf; white if it is white leaf; or ì$\perp$î meaning that the node does not exist. This approach can be used to represent grayscale images by storing quadrant values separately in another data structure like files.

3. Proposed Algorithm

The proposed *Multilevel Centroid* algorithm is a set redundancy based approach that is following the concept of the centroid method [2]. The algorithm was tested using two sets of medical images: CT and MRI and has shown an improved performance with respect to compression ratio.

3.1 Multilevel Centroid Technique

In this section, the *Multilevel Centroid* algorithm is introduced. The proposed algorithm applies the *centroid method* [2] on multilevel. Fig. 1 shows the multilevel centroid model. In that model, given a set of similar images $X = \{x_1, x_2, \text{Ö}..x_N\}$, the corresponding median image (median_1) is calculated. Applying the centroid method on the given input set, the difference_1 set (difference images at level 1) is obtained. Repeating the process recursively, the median_2 is calculated for the difference_1 set and applying the centroid method again, the difference_2 set is also obtained. The process stops when all levels are processed. The first level is exactly the same as the ordinary centroid method, which produces a difference image by subtracting a predicted image from the original one using *level_1* median image. The last difference_l set are compressed using any standard entropy compression method. Only *l* median images are needed and stored in order to compress individual images or used reversibly in the decompressing process.

Multilevel Centroid equations are as follows:

$$C_{i+1,j,l} = m_{l,i+1} + D_{i,j} - m_{l,i} \quad (6)$$

$$FD_{i+1,j,l} = D_{i+1,j} \text{ ñ } C_{i+1,j,l} \quad (7)$$

Where $C_{i+1,j,l}$ is the predicted value of position $i+1$ in image j at level l, $m_{l,i+1}$ is the pixel value of position $i+1$ in the median image at level l, $D_{i,j}$ is the value of position i in the input image j, and $FD_{i+1,j,l}$ is the final difference value between the predicted and input value at position $i + 1$ in image j at level l. For maximum l levels, the model equations are applied l times to compress any image.

Applying the centroid method on multilevel helps to obtain smaller values that are much closer to the zero than applying only one level such as in [2], thus reducing the dynamic gray level range that helps in improving the process of compression. Fig. 2 shows an example of applying two centroid levels on part of a CT brain image, where Fig. 2-a is a 16x16 8 b/p original image data taken from a CT brain image. Fig. 2-b is the 16x16 corresponding level_1 median image. By applying the centroid method in level_1, the output difference image is obtained using equations 4 and 5 and can be shown in Fig. 2-c with minimum and maximum values of ñ8 and 9 respectively. Fig. 2-d shows the level_2 median image while Fig. 2-e shows the output difference image after applying the centroid method in level_2 and it can be noticed that the minimum and maximum values in the second difference image are ñ3 and 3 respectively. This example shows that level_1 difference image requires 5 bits / pixel while level_2 difference image requires 3 bits / pixel. On the other hand, with entropy measures, level_1 difference image entropy = 2.7500 while level_2 difference image entropy = 1.5000.

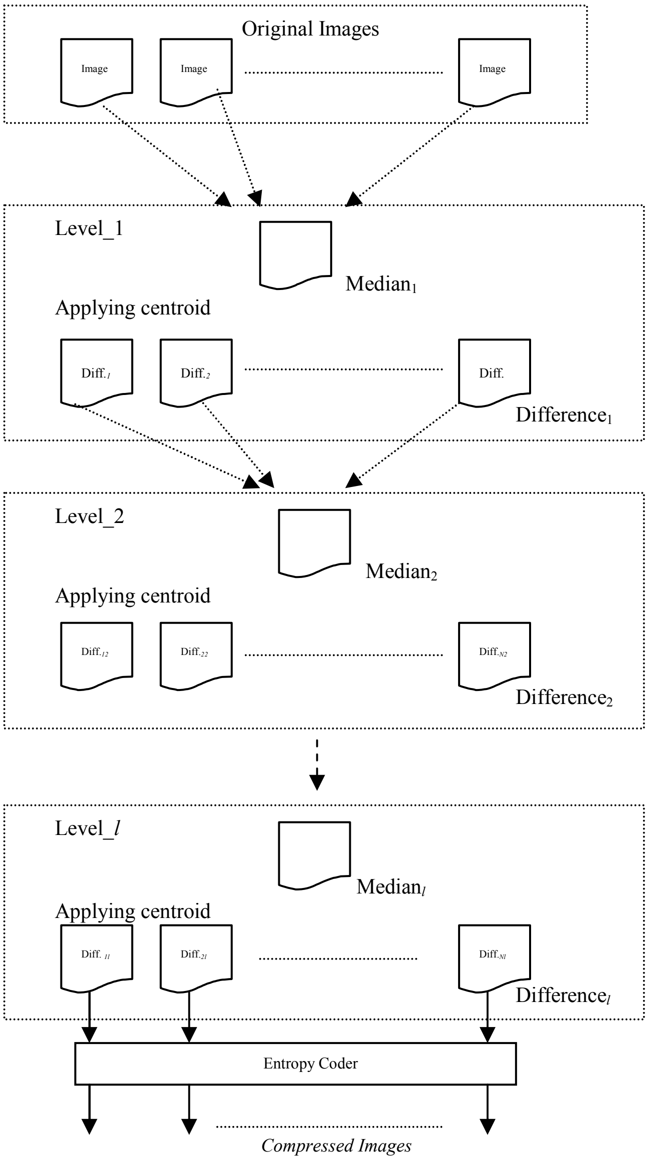

Fig. 1. Multilevel Centroid Compression Model

6	6	7	7	8	7	7	7	6	6	6	6	6	6	6	7
6	6	6	7	7	7	7	7	7	7	7	7	7	6	6	7
6	6	6	6	7	7	7	7	7	7	7	7	6	6	6	6
6	6	6	6	6	7	7	8	7	7	6	6	6	6	6	6
6	6	6	6	6	7	7	8	8	7	6	6	6	6	6	6
6	6	6	6	6	6	7	8	8	7	7	7	6	6	6	6
5	5	6	6	6	6	7	8	8	7	7	7	6	6	6	6
5	5	5	6	6	6	6	7	8	7	7	7	6	6	6	6
5	5	5	6	6	6	6	7	7	8	7	7	6	6	6	6
5	5	5	6	6	6	6	6	7	8	7	7	7	6	6	6
6	5	6	6	6	6	6	6	6	7	8	8	7	6	6	6
6	6	6	6	6	6	6	6	6	7	8	8	7	6	6	6
6	6	6	6	6	6	6	5	6	6	7	7	7	7	6	5
6	6	6	6	6	6	6	6	6	6	7	7	7	6	6	5
6	6	6	6	6	6	6	6	5	5	6	6	6	6	6	5
7	6	6	6	6	6	6	6	5	5	5	5	6	6	6	5

a- Original data

5	5	6	6	7	6	6	6	6	10	11	10	6	5	5	6
6	5	6	6	6	6	6	6	6	11	12	12	6	5	6	6
5	6	5	5	6	6	6	6	6	8	12	6	6	6	6	6
6	6	6	5	5	6	5	5	6	9	9	6	5	6	6	6
5	5	5	6	6	6	6	5	6	7	6	6	5	6	6	7
5	5	6	6	6	6	6	5	6	6	6	6	6	6	7	9
5	5	6	6	6	6	6	6	6	7	7	7	6	7	8	11
5	5	5	6	6	6	6	7	7	9	9	7	6	6	7	8
5	5	5	6	6	7	8	8	8	9	10	7	6	6	6	7
5	5	5	6	7	10	9	8	8	10	9	8	7	6	6	7
6	5	6	6	11	10	9	8	8	9	12	9	8	8	7	6
6	6	6	9	10	9	8	8	8	9	17	8	8	7	8	8
6	6	7	10	9	8	8	7	8	8	17	8	8	7	7	7
6	6	9	10	8	8	8	8	8	8	17	8	7	8	9	8
6	6	11	9	8	8	8	7	7	8	16	7	8	10	9	11
6	6	9	8	8	7	7	7	7	8	16	8	10	12	12	16

b- Level_1 median image

1	0	0	0	0	0	0	0	-1	-4	-1	1	4	1	0	0
0	1	-1	1	0	0	0	0	0	-5	-1	0	6	0	-1	1
1	-1	1	0	0	0	0	0	0	-2	-4	6	-1	0	0	0
0	0	0	1	0	0	1	1	-2	-3	-1	3	1	-1	0	0
1	0	0	-1	0	1	0	2	-1	-2	0	0	1	-1	0	-1
1	0	-1	0	0	0	1	2	-1	-1	0	0	-1	0	-1	-2
0	0	0	0	0	0	1	1	0	-2	0	0	0	-1	-1	-3
0	0	0	0	0	0	0	0	1	-3	0	2	0	0	-1	-1
0	0	0	0	0	-1	-1	1	0	0	-2	3	0	0	0	-1
0	0	0	0	-1	-3	1	1	1	-1	0	1	1	0	0	-1
0	0	0	0	-5	1	1	1	0	0	-2	3	0	-1	1	1
0	0	0	-3	-1	1	1	0	0	0	-7	9	-1	0	-1	0
0	0	-1	-3	1	1	0	0	0	0	-8	9	0	1	-1	-1
0	0	-3	-1	2	0	0	0	0	0	-8	9	1	-2	-1	0
0	0	-5	2	1	0	0	1	-1	-1	-7	9	-1	-2	1	-3
1	-1	-3	1	0	1	0	0	-1	-1	-8	8	-1	-2	0	-5

c- Level_1 difference image (entropy = 2.7500)

0	0	0	0	-1	1	0	0	0	-4	-1	1	4	1	0	-1
-1	1	-1	0	0	0	0	0	0	-5	-1	0	6	0	-1	0
0	-1	1	0	0	0	0	0	0	-2	-4	6	0	0	0	0
-1	0	0	1	0	-1	1	0	-1	-3	0	3	1	0	0	0
0	0	0	-1	0	0	0	1	-1	-1	1	0	1	-1	0	-1
1	0	-1	0	0	0	0	1	-1	0	0	0	0	0	-1	-2
1	0	-1	0	0	0	0	0	0	-1	0	0	1	-1	-1	-3
1	0	0	-1	0	0	0	-1	0	-2	0	2	1	0	-1	-1
1	0	0	-1	0	-1	-1	0	0	-1	-1	3	1	0	0	-1
1	0	0	-1	-1	-3	1	1	0	-2	1	1	1	1	0	-1
1	1	0	0	-5	1	1	1	0	-1	-3	3	1	0	1	1
1	0	0	-3	-1	1	1	0	0	-1	-8	9	0	1	-1	0
2	0	-1	-3	1	1	0	1	-1	0	-9	9	0	1	0	0
3	0	-3	-1	2	0	0	0	0	0	-9	9	1	-1	-1	1
3	0	-5	2	1	0	0	1	0	-1	-8	9	-1	-2	1	-2
3	0	-3	1	0	1	0	0	0	-1	-8	8	-1	-2	0	-4

d-Level_2 median image

1	-1	0	0	1	-2	1	0	-1	1	0	0	0	0	0	1
1	-1	0	1	-1	0	0	0	0	0	0	0	0	0	0	1
1	-1	0	0	0	0	0	0	0	0	0	0	-1	1	0	0
1	-1	0	0	0	1	-1	1	-2	1	-1	1	0	-1	1	0
1	-1	0	0	0	1	-1	1	-1	-1	0	1	0	0	0	0
0	0	0	0	0	0	1	0	-1	-1	1	0	-1	1	0	0
-1	1	1	-1	0	0	1	0	-1	-1	1	0	-1	1	0	0
-1	1	0	1	-1	0	0	1	0	-2	1	0	-1	1	0	0
-1	1	0	1	-1	0	0	1	-1	1	-2	1	-1	1	0	0
-1	1	0	1	-1	0	0	0	1	0	-2	1	0	-1	1	0
-1	0	1	0	0	0	0	0	0	1	0	-1	-1	0	1	0
-1	1	0	0	0	0	0	0	0	1	0	-1	-1	0	1	0
-2	2	0	0	0	0	0	-1	2	-1	1	-1	0	0	-1	0
-3	3	0	0	0	0	0	0	0	0	1	-1	0	-1	1	-1
-3	3	0	0	0	0	0	0	-1	1	1	-1	0	0	0	-1
-2	1	1	0	0	0	0	0	-1	1	0	0	0	0	0	-1

e-Level_2 difference image (entropy = 1.5000)

Fig. 2 Multilevel Centroid example on a 16x16 CT brain image

4. Experimental results

The performance of the proposed algorithm was tested using 10 512x512 brain CT images, and 10 256x256 MRI brain images. Figures 3 and 4 show the test images used of each group. Both group of images are 8 bits/pixel gray-level images, each set of images are similar to each other, and are the same test images that were used by [2] in testing the centroid method. They were collected at M.D. Anderson Cancer Center in Houston, Texas out of a small image database with 51 CT brain images and 57 MRI brain images, from random patients of both sexes, different ages, and a variety of pathological conditions [2]. Each method was tested in combination with the most widely used entropy-based compression techniques [12]: Huffman encoding and Arithmetic coding.

The compression ratio is defined as

$$C = \text{original image size} / \text{compressed image size} \tag{8}$$

The multilevel centroid algorithm has been tested on different levels, starting from one level which is identical to centroid method [2] and up to three levels, that to demonstrate the performance of the new algorithm in different levels.

4.1 CT Experiment Results

Table 1. shows the experimental results of the average compression ratio achieved when applying the traditional centroid method [2] and the proposed multilevel centroid algorithm using two and three levels on the ten CT brain images shown in Fig. 3 using Huffman and Arithmetic entropy encoders. It can be shown that the compression ratio improvement range of *level-2* is about 38% to 40% while *level-3* is about 19.5% to 21% in comparison to the centroid method[2].

The improvement of using the multilevel centroid method is 160% and 127% for both *level_2* and *level_3* respectively in comparison to standard Huffman compression method. Meanwhile, the improvement in comparison to Arithmetic compression method, is about 121% and 88% for *level_2* and *level_3* respectively.

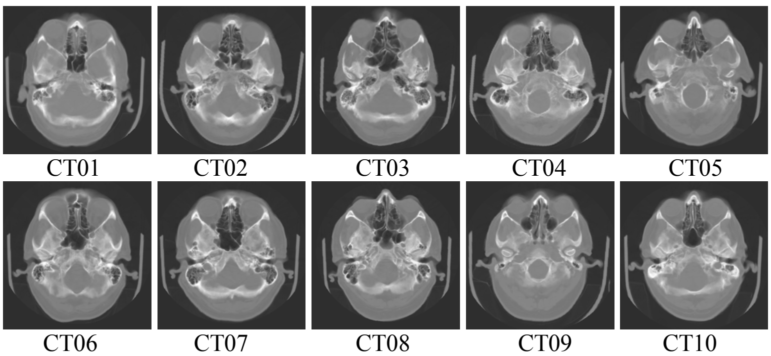

Fig. 3. CT brain test images

	Huffman	Arithmetic
	C.R	C.R
Regular compression method	1.3790 : 1	1.7010 : 1
Centroid method [2] (1 level)	2.5940 : 1	2.6734 : 1
Multilevel Centroid (2 levels)	3.5824 : 1	3.7521 : 1
Multilevel Centroid (3 levels)	3.1310 : 1	3.1978 : 1

Table 1. Average compression ratio of CT images

4.2 MRI Experimental Results

Table 2 shows the experimental results of the average compression ratio achieved when applying the traditional centroid method [2,17] and the proposed multilevel centroid algorithm using two and three levels on the ten MRI brain images shown in Fig. 4 using Huffman and arithmetic entropy encoders. It can be shown that the compression ratio improvement range is about 6.6% to 9.8% for *level-2* while for *level-3* no improvement has been achieved. The achieved compression ratio is small with respect to the CT images due to the low signal to noise ratio of the MRI images.
The improvement of using the multilevel centroid method is 36% and 22% for both *level_2* and *level_3* respectively in comparison to standard Huffman compression method. Meanwhile, the improvement in comparison to Arithmetic compression method is about 26% and 13% for *level_2* and *level_3* respectively.

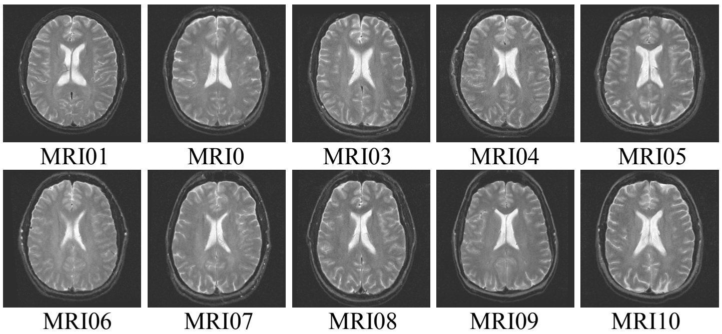

Fig. 4. MRI brain test images

	Huffman	Arithmetic
	C.R	C.R
Regular compression method	1.2940 : 1	1.4040 : 1
Centroid method [2] (1 level)	1.6500 : 1	1.6097 : 1
Multilevel Centroid (2 levels)	1.7589 : 1	1.7678 : 1
Multilevel Centroid (3 levels)	1.5786 : 1	1.5826 : 1

Table 2. average compression ratio of MRI images

5. Discussions and Conclusions

Testing the new algorithm on CT and MRI test images has shown that applying the centroid method recursively on more than one level improves the average compression ratio. Level_2 gives the best results while level_3 does not give further improvements. The objective of the algorithm is to reduce the dynamic range of gray-level values to improve compression performance; this is achieved in level_2 difference image values, while in level_3 the difference values become more sensitive to change, as the prediction model does not predict close values to the original data, thus obtaining more overhead than level_2.

References

1. D. Clunie, Lossless compression of grayscale medical images ñ Effectiveness of traditional and state of the art approaches, in Proc. SPIE-Medical Imaging, Vol. 3980, 2000.
2. K. Karadimitriou et al., Centroid method for compressing sets of similar images, Pattern Recognition Letters, vol.19, no.7 (1998) pp.585-593.
3. M. Manouvrier et al., Quadtree representations for storage and manipulation of clusters of images, Image and Vision Computing 20 (2002) 513-527.
4. R.A.Finkel, J.Bentley, Quadtrees: a data structure for retrieval on composite keys, Acta Informatica 4 (1) (1974) 1- 9.
5. H. Samet, The quadtree and related hierarchical structures, Computer Surveys 16 (2) (1984) 187-260.
6. H.Samet, R.E. Webber, Hierarchical data structures and algorithms for computer Graphics and Applications 8 (5) (1988) 48-68.
7. H. Lu, B.-C. Ooi, K.-L. Tan, Efficient Image Retrieval by Color Contents, International Conference on Applications of Database, Sweden, LNCS, vol. 819, Springs, N.Y., 1994.
8. K. Karadimitriou and J.M. Tyler, The Min-Max Differential Method for Large-Scale Storage and Compression of Medical Images, in Proc of Annual Molecular Biology and Biotechnology Conference, Baton Rouge, Feb. 1996.
9. K. Karadimitriou and J.M. Tyler, Min-Max Compression Methods for medical image databases. ACM SIGMOD Record 26, (1997) 47-52.
10. J.P. Cheiney, A. Touir, FI-Quadtree: a new data structure for content-oriented retrieval and fuzzy search, Proceedings of the Second Symposium on Spatial Databases (SSDí91), 1991.
11. G. Jomier, M. Manouvrier, M.Rukoz, Storage and management of similar images, Journal of the Brazilian Computer Society (JBCS) 3 (6) (2000) 13- 26.
12. K. Skretting, J.H.Husÿ y, S.O.Aase, Improved Huffman Coding Using Recursive Splitting, NORSIG-99 Conference on Signal Processing and Signal Compression, 9-11 sep, 1999.
13. M. Vassilakopoulos, Y. Manolopoulos, Dynamic inverted quadtree- a structure for pictorial databases, information systems, Special Issue on Multimedia Information Systems 20 (6) (1995) 483-500.
14. M. Vassilakopoulos, Y. Manolopoulos, K. Economou, Overlapping quadtrees for the representation of similar images, Image and Vision Computing 11 (5) (1993) 257-262.
15. T. Tzouramanis, M. Vassilakopoulos, Y. Manolopoulos, Overlapping Linear Quadtrees: a Spatio-Temporal Access Method, ACM GISí98, Washington DC, 1998.
16. T. ñW. Lin, Compressed quadtree representations for storing similar images, Image and Vision Computing 15 (1997) 833-843.
17. K. Karadimitriou, Set Redundancy, the Enhanced Compression Model, and Methods for Compressing Sets of Similar Images, Ph.D. Dissertation, Louisiana University, Aug. 1996.